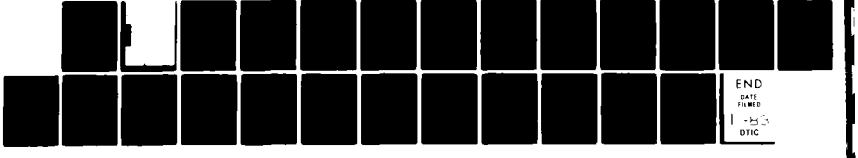
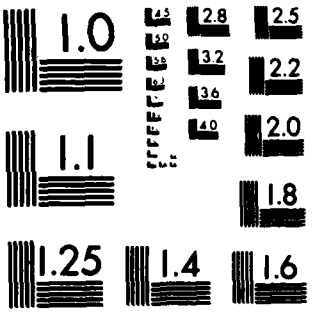
AD-A121 602 THE A-7E SOFTWARE REQUIREMENTS DOCUMENT: THREE YEARS OF 1/1
CHANGE DATA(U) NAVAL RESEARCH LAB WASHINGTON DC
L J CHMURA ET AL. 08 NOV 82 NRL-MR-4938 SBI-AD-E000 506

UNCLASSIFIED F/G 9/2 NL



END
DATE FILMED
1-83
DTIC



SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER NRL Memorandum Report 4938	2. GOVT ACCESSION NO. A121 602	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) THE A-7E SOFTWARE REQUIREMENTS DOCUMENT: THREE YEARS OF CHANGE DATA		5. TYPE OF REPORT & PERIOD COVERED Interim report on a continuing NRL problem. 6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) L.J. Chmura and D.M. Weiss		8. CONTRACT OR GRANT NUMBER(s)
9. PERFORMING ORGANIZATION NAME AND ADDRESS Naval Research Laboratory Washington, DC 20375		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS 61153N; RR014-09-41; 75-0199-0-2
11. CONTROLLING OFFICE NAME AND ADDRESS		12. REPORT DATE November 8, 1982 13. NUMBER OF PAGES 23
14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office)		15. SECURITY CLASS. (of this report) UNCLASSIFIED 15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Software measurement
Software change data
Software change analysis
Software development methodology evaluation
Software requirements

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

A major product of the Naval Research Laboratory's Software Cost Reduction project is the software requirements document for the Navy's A-7E aircraft operational flight program. The document, which was first published in November 1978, is intended to serve as a model for specifying complex software systems. We have monitored all changes to the document. The change data have consistently suggested that the specification has several desirable qualities, for example, it is easily maintained and is remarkably free of inappropriate implementation detail.

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE
S/N 0102-014-6601

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

CONTENTS

THE A-7E SOFTWARE REQUIREMENTS DOCUMENT: THREE YEARS OF CHANGE DATA

INTRODUCTION

General

Basili and Weiss (1981) as well as Fryer and Weiss (1981) have reported earlier on change data being collected to evaluate the software requirements document (SRD) for the A-7E aircraft (Heninger et al. 1978). The SRD is a product of the Naval Research Laboratory's Software Cost Reduction (SCR) project. Here, we present a more recent picture of the data and, where possible, compare with other published data. In preparation for this paper, we have reexamined all submitted change report forms for consistency and completeness of information. This has resulted in the reclassification of some earlier data with variable effects on the earlier published observations.

The remaining three sections of this introduction review the SCR project, the SRD, and goal-directed data collection for those who are unfamiliar with these topics.

Software Cost Reduction (SCR) Project

Since January 1978, personnel at the Naval Research Laboratory (NRL) and the Naval Weapons Center (NWC) have been redeveloping version NWC-2 of the operational flight program (OFP) for the A-7E aircraft using such software engineering techniques as information hiding (Parnas 1972), abstract interfaces (Parnas 1977), cooperating sequential processes (Dijkstra 1968), and resource monitors (Hoare 1974). This A-7E OFP redevelopment is currently referred to as the Software Cost Reduction (SCR) project.

The A-7E OFP is part of the Navigation/Weapon Delivery System on the A-7E aircraft. It receives input data from sensors, cockpit switches, and a panel with which a pilot keys in data. It controls several displays in the cockpit and positions several sensors. In all, twenty-two devices are connected to the TC-2; examples include an inertial measurement set providing velocity data and a head-up display. The head-up display projects symbols into a pilot's field of view, so that he sees them overlaying the world ahead of the aircraft. The OFP calculates navigation information such as present position, speed, and heading; it also controls weapon delivery, giving a pilot steering cues and calculating when to release weapons.

The A-7E OFP is an operational Navy program with severe memory and execution-time constraints. The code consists of about 12,000 assembler language instructions for the IBM System 4 PI model TC-2 computer. The TC-2 has 16K bytes of memory.

An earlier version of this paper appeared in the proceedings of the Avionics Panel (AVP) symposium on 'Software For Avionics', which was sponsored by NATO's Advisory Group For Aerospace Research and Development (AGARD) and held at the Hague, Netherlands, 6-10 September 1982.

Manuscript approved September 16, 1982.

The goals of the SCR project are (1) to demonstrate the feasibility of using the selected software engineering techniques in developing complex, real-time software and (2) to provide the Navy with a model for the design of avionics software. One of the reasons for choosing to redevelop the A-7E OFP is the challenge of showing that any memory or execution-time overhead incurred by using the software engineering techniques is not prohibitive for such real-time programs. A second reason is that maintenance personnel at NWC feel that the OFP is difficult to change; the claim for the selected software engineering techniques is that they facilitate the development of software that is easy to change.

The A-7E SRD (Heninger et al. 1978) is the first major product of the SCR project. More recent products of ongoing software design include a guide to SCR software modules (Britton and Parnas 1981), interface specifications for the device interface module (Parker et al. 1980), specifications for the function driver module (Clements 1981), specifications for the extended computer module (Britton et al. 1982), and specifications for the shared services module (Clements 1982).

The projected completion date for the SCR project is September 1985. As of the end of 1981, approximately 10 man-years of technical engineering effort have been expended on the redevelopment.

A-7E Software Requirements Document (SRD)

The SRD is an attempt to provide a complete and concise description of the A-7E's OFP requirements. No other such requirements description exists despite the existence of a working OFP. The SRD is also an attempt to improve upon present methods of specifying software requirements. Four principles motivate the document: (1) state questions before trying to answer them, (2) separate concerns, (3) be as formal as possible, and (4) organize according to OFP outputs (Heninger 1980). The resultant organization is shown in figure 1.

Personnel at NWC who are maintaining the current A-7E OFP have been active consultants and reviewers of the document, both prior to initial publication and subsequently. The document has been under configuration control since it was first published in November 1978.

As of the end of 1981, approximately 2.5 man-years of technical engineering effort has been expended on the document: approximately 1.5 man-years on producing the original document and the remaining 1 man-year on changing and issuing three updates.

Goal-Directed Data Collection

The opportunity to apply recent software engineering technology liberally to the development of a complex system is rare. Though true evaluation of the result must wait until delivery of the software, we believe it would be unfortunate to wait until then. From the start of the redevelopment, we have collected data to permit evaluation of the project and the products in the interim.

The data collection methodology being used is goal directed (Weiss 1981).

Briefly, it consists of the following five elements.

1. Identify Goals. The data collection effort can be geared to determine how well the goals for a product or process are met. Twelve goals drive data collection for the SRD. Six are original goals for the document (Heninger 1980).

   1. Specify external behavior only.
   2. Specify constraints on the implementation.
   3. Be easy to change.
   4. Serve as a reference tool.
   5. Record forethought about the system life cycle.
   6. Characterize acceptable responses to undesired events or errors.

   Three other goals apply to requirements documents in general.

   7. Be correct.
   8. Promote detection and correction of errors.
   9. Be useful.

   Another three concern the software development process.

   10. Discover effective ways of finding errors.
   11. Characterize changes.
   12. Characterize errors.

2. Determine Questions Of Interest From The Goals. The answer to each question can then help measure how well a stated goal has been obtained. For example, given the goal that the SRD be easy to change, a reasonable question would be:

   Are changes confined to a single section of the document?

   Such a question suggests that data be collected on how many sections were modified for each requirements change.

   Figure 2 is a complete list of questions for the SRD.

3. Develop A Data Collection Form. This is an iterative process. The form is best tailored to the product being studied and to the questions of interest. Figure 3 is the change report form (CRF) currently used to request and record resolution of changes to the SRD.

4. Develop Data Collection Procedures. Procedures for collecting change data are best part of normal configuration control procedures.

5. Validate And Analyze The Data. Reviews and analyses of the accumulating data are concurrent with software development. We have found that validation of the data should occur weekly. The validation should include examining the forms for completeness and consistency. Sometimes it is necessary to interview the originator or the person making the change to obtain omitted data or to resolve problems. Validation of A-7E requirements change forms has several times detected incomplete changes.

A-7E SOFTWARE REQUIREMENTS CHANGE DATA

Background

There are 343 change report forms (CRFs) that were generated against the SRD from November 1978, when it was first published, through December 1981. Only 284 of the requested changes have been resolved (i.e., carefully analyzed and formally accepted or rejected) by the end of 1981. Specifically, 276 have been accepted; 8 rejected (only 2 of which were rejected because the change was deemed not worth the effort.) We have validated the 276 accepted CRFs. We feel most confident in using the data to answer the following seven questions about the SRD from figure 2.

1. Is the requirements document easy to change?
2. Is it clear where a change has to be made?
3. Are changes confined to a single section?
4. What use of the document reveals the most errors?
5. How many errors are found in the document?
6. What kinds of errors are contained in the document?
7. Which sections have the most errors?

It is important to understand what we believe properly constitutes a "change." For the SRD, we define a change to be an alteration to a baselined version. We consider two changes to be the same if they have the same cause and are written against the same version of the SRD. As an example, connecting a new device to the A-7E computer might require numerous updates to the SRD, but all the updates would make up a single change. A change that is a completion or correction of a prior change is a separate change.

Occasionally, a CRF is submitted containing two changes. When this happens, we generate a second CRF and separate the two changes onto two forms. For example, if the following correction was submitted on a CRF, we would generate a second CRF. The first would address just the ambiguity, the second would address the misspelling.

> "The last sentence of the description is ambiguous. Replace it with .... Note also that the word descripter is misspelled."

Conversely, if two CRFs are submitted that are different parts of the same change, we merge the two CRFs.

We consider that there are two classes of change data: error corrections and non-error corrections. An error correction is either an original error correction (i.e., the first correction of the error) or a completion or correction of a previous change. In terms of figure 3, the CRF for an error correction would have one of the two first two boxes in item 4 marked and the second page filled in. Hereafter, we use the term error to refer to error corrections and the term modification to refer to non-error corrections. The following table shows the number of errors and modifications reported and resolved each year.

	YEAR CHANGE ACCEPTED				
Change Class	1978	1979	1980	1981	1978-1981
Error	4	72	97	74	247
Modification	0	5	13	11	29
Total:	4	77	110	85	276

The small number of modifications can be explained by the fact the the SCR project is a faithful redevelopment of version NWC-2 of the A-7E OFP.

There are eight classes of requirements errors: clerical, ambiguity, omission, inconsistency, incorrect fact, information put into wrong section, implementation fact included, and other (see item 7 in figure 3). The classification scheme is generally in terms of cause; that is, we classify errors according to their causes, not according to their symptoms. Accordingly, we use the following definitions for the different error classes.

Clerical Error: An error resulting from a mechanical transcription process from one medium to another (e.g., keypunch error when copying from handwritten to computer-processable form).

Ambiguity: An error resulting from an author's inability to distinguish clearly among several alternatives. Note that an ambiguity might result from unavailability of information, carelessness, or other reasons.

Omission: An error resulting from an author knowing necessary information but not including it in the document.

Inconsistency: An error resulting from authors of two or more different sections of the document believing contradictory statements. The result is that two different parts of the document contradict each other.

Incorrect Fact: An error resulting from an author having the wrong information. The result is that the document contradicts other sources of information.

Information Put Into Wrong Section: An error resulting from improper separation of concerns, such as including a description of the data available from a radar in the same section as a description of the computer interrupt structure.

Implementation Fact Included: An error resulting from an author including information about how to implement a requirement.

Other: An error resulting from some cause other than those specified in the foregoing.

Note that determining the cause, and hence the class, of an error sometimes

requires the help of the authors of the document. Furthermore, there are some errors for which the proper class is not completely clear. To reduce the number of such situations, we train all data analysts to use the same classification criteria and use more than one analyst to review questionable cases. We estimate that fewer than 5% of the changes may be improperly classified.

The statistics we present on the effort required to design changes follow from the times supplied in item 5 of the requirements CRF. The times are those that technical persons expend on understanding and specifying (i.e., designing) changes in sufficient detail so that an editor or typist can update the document maintained on a word processor. Thus, editorial and secretarial time are not included.

We classify the effort expended in designing changes as follows:

Class	Design Effort (e)
Trivial (T)	$0 < e \leqslant 1$ work hour (wh)
Easy (E)	$1\ \text{wh} < e \leqslant 1$ work day (wd)
Moderate (M)	$1\ \text{wd} < e \leqslant 1$ work week (ww)
Difficult (D)	$1\ \text{ww} < e \leqslant 1$ work month (wm)
Formidable (F)	$1\ \text{wm} < e$

It is interesting that, of the 276 accepted changes, none have been Difficult (D) changes and only 2 have been Formidable (F) changes.

Is the document easy to change?

Figure 4 is the distribution of effort for designing most of the changes to the SRD. Excluded are changes to correct clerical errors because such changes typically involve simple fixes such as correcting misspellings or punctuation. Excluded also are change completions, though the efforts for change completions have been added to the efforts reported for the corresponding original changes. The fact that 94% of the changes required a day or less to design, together with the fact that only two requested changes were rejected because the efforts to make them seemed unjustified, suggest that the SRD is indeed easy to change. What's more, the contrasts between figure 5 and figure 6 suggest that recent changes tend to be even easier than earlier changes.

Figure 7 is an alternative for figure 4. Changes to correct clerical errors are again excluded, but change completions are included. Because the majority of the change completions continued the work initiated by two formidable changes, the distribution of figure 7 is close to that of figure 4.

Actually, the general form of figures 4 through 7 is to be expected; it seems to be characteristic of accepted changes in general. It is quite similar to figure 8, a distribution of effort to design software changes for a NASA software development project (Weiss 1981). It is also similar to the distribution of working time to correct errors during test and integration as presented by Shooman and Bolsky (1975). The simple fact of the matter may be

that a software project simply cannot survive if personnel tie themselves up with making too many time-consuming changes; potentially difficult or formidable changes will tend to be rejected unless there is no other way out. What is remarkable about figures 4 through 7 is that only two requested changes have been rejected because they were deemed not worth the effort.

As a note of caution, recall that the SCR project is different from many software-production projects in that it is a faithful redevelopment of an existing OFP. The variety of changes for the SRD is less than for many such documents. For example, the OFP interfaces have not changed because of the introduction of new interfacing equipment.

Is it clear where a change has to be made?

Of the 276 requirements changes, 70 correct clerical errors. Of the remaining 206 changes, only 23 (11%) are change completions. This small percentage by itself suggests it is clear where requirements changes have to be made. Another measure suggests the same: Of the 206 changes, only 35 (17%) required examination of more sections than were changed.

Figure 9 shows the yearly percentage of changes that required examination of more sections than the number actually changed. The trend is not encouraging. Recent changes, which tend to be easier as noted in section 2.2, are nevertheless requiring the examination of more material. One explanation for the trend may be the fact that the original authors of the requirements document no longer design most of the changes; persons less expert in the document have taken their places. Another explanation may be simply that recent changes are more subtle than earlier changes. We will have to wait to see which explanation applies. If the trend slows or stops in the future, then the first explanation would seem to hold. If the trend continues, then the second would seem more valid.

Are changes confined to a single section?

There are 183 changes excluding change completions and clerical errors. If we combine change completion data with the corresponding original changes, then 40 (22%) of the 183 changes involve more than one section. Figure 10 plots this statistic over the years. Together these numbers indicate that, early on, changes tended only slightly to be confined to one section. More recently, confinement to one section tends to be the rule.

What use of the document reveals the most errors?

Of the 74 errors corrected in 1981, 18 are clerical. The distribution of ways in which the remaining 56 nonclerical errors have been detected is given in figure 11. The distribution clearly shows that, in 1981, the SCR project was heavily into design. Figure 12 shows that design activity has been the primary way in which errors have been detected. Although not surprising statistics, they are nevertheless encouraging because they indicate that the SRD is being heavily used by designers. Apparently, the document is meeting two of the goals stated for it: (1) that it be useful and (2) that it serve as a reference tool.

The cumulative distribution of figure 12 should be of special interest toward the end of the SCR project. If the requirements document is of high quality and if the approach being taken to software development is successful, then relatively few nonclerical requirements errors should be reported as a result of testing. In other words, the distribution of figure 12 should retain its same general shape.

How many errors are found in the document?

There are 247 requirements errors that were corrected from 1978 through 1981, 70 (28%) of which are clerical. This leaves 177 nonclerical errors, which seems a small number considering that the SRD contains approximately 600 pages. The error-per-page ratio is only 0.30.

Bell and Thayer (1976) report on 972 problems with a B-5 level software specification that comprised approximately 2500 pages. About 50 (5%) are problems of new or changed requirements, a type that cannot occur in the A-7E OFP redevelopment. The remaining 922 problems yield a problem-per-page ratio of 0.37, which is not very different from the above ratio for the SRD. But the problems reported by Bell and Thayer are the result largely of two formal requirements reviews conducted during system development. They make up some fraction of the total number of problems found. Thus, the actual problem-per-page ratio is likely much greater than 0.37.

What kinds of errors are contained in the document?

The distribution of error classes in 1981 is shown in figure 13; in 1979, figure 14; cumulatively, figure 15. Except for some variation in the percentage of inconsistencies, there is little difference in the distributions. The data suggest that the requirements document is much more precise (has relatively few ambiguities) and somewhat more consistent than it is complete and correct. Perhaps most remarkable is that there are no errors of the type "inclusion of implementation facts." The authors seem to have succeeded extremely well in their goal of specifying external behavior only.

It is also interesting to note that the percentage of clerical errors found has remained constant with time. Of the 70 clerical errors so far found, only two were introduced by earlier changes.

Which sections have the most errors?

There are 177 nonclerical errors that were corrected from 1978 through 1981. Twenty one (12%) have involved updating more than one major section of the SRD. Figure 16 shows the impact of the errors. The distribution is close to that reported by Basili and Weiss (1981) for nonclerical errors corrected from November 1978 through February 1980. Section 2 (Input-Output Data Items), section 4 (Software Functions), and the dictionary continue to contain the most errors. A major difference, however, is that section 4 and the dictionary now both show more problems than does section 2. This trend is most likely the result of a shift in software design activity from heavy use of section 2 information early on to heavy use of section 4 information later.

Figure 17 shows the ratio of nonclerical errors per page by section of the SRD. Except for the ratios for section 3, section 4, and the dictionary, all

ratios are close to the ratio given previously for the entire SRD. The large ratio for the dictionary might be explained by the dense nature of that section; it is mostly a long list of rather short definitions. What's more, the dictionary terms are used extensively throughout the SRD. As a result, the dictionary is frequently involved in multisection changes. The ratio for section 4 suggests that its material is either intrinsically difficult or that the specification approach is slightly flawed. The rather small ratio for section 3 is a pleasant surprise given that the section defines the numerous OFP operational modes as well as the permissable mode transitions.

CONCLUSIONS

We have two objectives in monitoring changes to the SRD. The first is to test the feasibility of goal-directed data collection. The second is to measure the success of the document's authors in meeting their objectives. This second objective is important because one of the main goals of the SCR project is to produce a model for engineering OFP software. The SRD is an important part of that model.

Our work has shown that goal-directed data collection can be feasibly integrated with traditional configuration control activities. The major difficulty is that there must be constant careful attention to ensure accurate information; that is, validation must be concurrent with data collection.

Our analysis of the three years of change data suggests that the authors of the SRD have successfully met some major objectives. In particular:

1. The document seems to be easily maintained, even though a growing percentage of recent changes requires the examination of more sections than those that must be changed.

2. The document seems to be well structured. Changes tend to be confined to single sections.

3. Based on comparison with published data, relatively few errors have been found to date.

4. The document is remarkably free of ambiguities and inappropriate implementation details.

5. The dictionary and specification of software functions in section 4 of the document are the most error prone sections of the SRD. The specification of system modes and mode transitions in section 3 is remarkably error free.

6. The document is a living document. It has been heavily used during design.

These are of course interim results, and data collection will continue throughout the SCR project. Nevertheless, the change data from the start has consistently supported this positive evaluation of the SRD (see Basili and Weiss 1981). There do not appear to be any significant trends in the data that suggest the conclusions will change.

0. Introduction: A description of the document organization, an abstract for each section, and a guide to the notation used.

1. Distinguishing Characteristics of the TC-2 Computer

2. Input and Output Data Items: Description of the information received and transmitted by the computer, organized according to device, one subsection per device connected to the computer.

3. Modes of Operation: States of the program corresponding to aircraft operating conditions.

4. Time-independent Description of A-7 Software Functions: Each function description characterizes one or more output date items and specifies the conditions under which they are updated.

5. Timing Requirements: Timing requirements for all functions described in section 4.

6. Accuracy Constraints on Software Functions

7. Undesired Event (UE) Responses: Desired behavior of the system when undesired events occur.

8. Required Subsets: Useful subsets of the system obtainable by omitting parts of the code.

9. Possible Changes: Possible future modifications to the OFP.

10. Glossary: Glossary of acronyms and technical terms used by the A-7 community.

11. References

Indices: Alphabetical indices to data item descriptions, mode overviews, and functions.

Dictionary: Definitions of standard terms used in the mode (section 3) and function (section 4) descriptions.

Fig. 1 — Sections of the A-7E software requirements document

1. Is externally-visible behavior only specified without implying a particular implementation?

2. Are the appropriate external interfaces specified?

3. Are the external interfaces specified correctly?

4. Is the document easy to change?

5. Is it clear where a change has to be made?

6. Are the changes likely to occur predicted correctly?

7. Are changes confined to a single section?

8. Is the proper set of undesired events described?

9. Is the notation used unambiguous?

10. Which sections have the most errors?

11. Where do the most changes have to be made?

12. Which type of tables has the most errors?

13. Does the document contain unnecessary information?

14. What use of the document reveals the most errors?

15. Are sections 3 (Modes) and 4 (Functions) consistent with each other?

16. Is the dictionary complete, correct, and consistent with the rest of the document, and will it remain so?

17. Which subsections of sections 2 (Data Items), 3 (Modes), and 4 (Functions) are most error-prone?

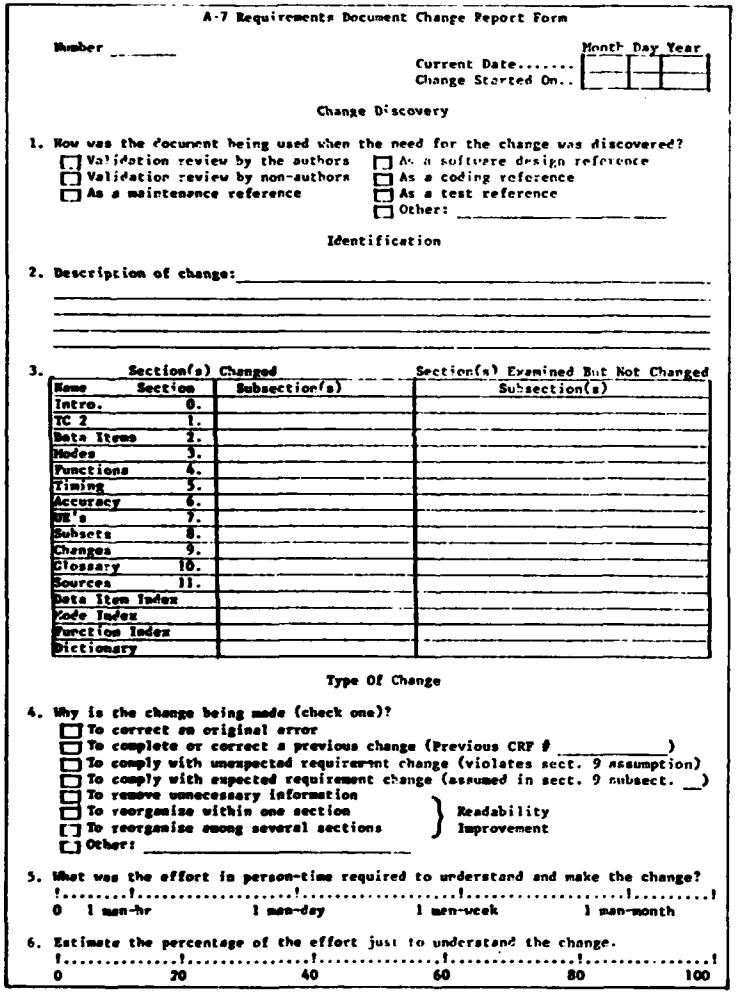
18. How is the document being used?

19. Why are changes being made?

20. How many errors are found in the document?

21. What kinds of errors are contained in the document?

Fig. 2 — Questions underlying the change report form (CRF) for the A-7E software requirements document



A-7 Requirements Document Change Report Form

Number _____

Current Date..... Month Day Year
Change Started On..

Change Discovery

1. How was the document being used when the need for the change was discovered?
- ☐ Validation review by the authors
- ☐ Validation review by non-authors
- ☐ As a maintenance reference
- ☐ As a software design reference
- ☐ As a coding reference
- ☐ As a test reference
- ☐ Other: _____

Identification

2. Description of change: _____

3.

	Section(s) Changed		Section(s) Examined But Not Changed
Name	Section	Subsection(s)	Subsection(s)
Intro.	0.		
TC 2	1.		
Data Items	2.		
Modes	3.		
Functions	4.		
Timing	5.		
Accuracy	6.		
UE's	7.		
Subsets	8.		
Changes	9.		
Glossary	10.		
Sources	11.		
Data Item Index			
Code Index			
Function Index			
Dictionary			

Type Of Change

4. Why is the change being made (check one)?
- ☐ To correct an original error
- ☐ To complete or correct a previous change (Previous CRF # _____)
- ☐ To comply with unexpected requirement change (violates sect. 9 assumption)
- ☐ To comply with expected requirement change (assumed in sect. 9 subsect. __)
- ☐ To remove unnecessary information
- ☐ To reorganize within one section
- ☐ To reorganize among several sections
- ☐ Other: _____

} Readability Improvement

5. What was the effort in person-time required to understand and make the change?

0 | 1 man-hr | 1 man-day | 1 man-week | 1 man-month

6. Estimate the percentage of the effort just to understand the change.

0 | 20 | 40 | 60 | 80 | 100

FOR ERROR CORRECTIONS ONLY

Type Of Error

7. How is the error best characterized (check one)?
- ☐ Clerical
- ☐ Ambiguity
- ☐ Omission
- ☐ Inconsistency
- ☐ Incorrect Fact
- ☐ Information put in wrong section
- ☐ Implementation fact included
- ☐ Other: _____

} Inappropriate Fact

For Section 2 (Data Item) Errors

8. Which part(s) of the subsection were incorrect (check all that apply)?
- ☐ Hardware
- ☐ Description
- ☐ Value Characteristics
- ☐ Instruction Sequence
- ☐ Data Representation
- ☐ Comment

2

For Section 3 (Modes) and Section 4 (Functions) Errors

How is the error best characterized (check all that apply)?

9. Section 3
- ☐ Mode Transition Error
- ☐ Mode Condition Error
- ☐ Mode Overview Error

3

10. Section 4
- ☐ Applicable Mode List Error
- ☐ Output Data Item Error
- ☐ Output Description Error
- ☐ Event Table Error
- ☐ Selector Table Error
- ☐ Condition Table Error
- ☐ Inconsistent or incomplete table

4

11. ☐ Corresponding errors were made in both sections 3 and 4.

For Errors Involving Dictionary Items

12. How is the error best characterized (check all that apply)?
- ☐ Incorrect item
- ☐ Incomplete item
- ☐ Dictionary inconsistent with usage elsewhere

DICT.

For Notational Errors

13. Which notation type was in error?
- ☐ /input/
- ☐ //output//
- ☐ $value$
- ☐ !item!
- ☐ *mode*
- ☐ operator
- ☐ event
- ☐ simple condition
- ☐ compound condition

14. How is the error best characterized?
- ☐ incorrect grouping
- ☐ incorrect symbol
- ☐ incorrect value

NOT.

Comments

15. Please give any information that may help understand the change and its cause.

_____ Name _____ Date

Fig. 3 — The change report form (CRF) for the A-7E software requirements document

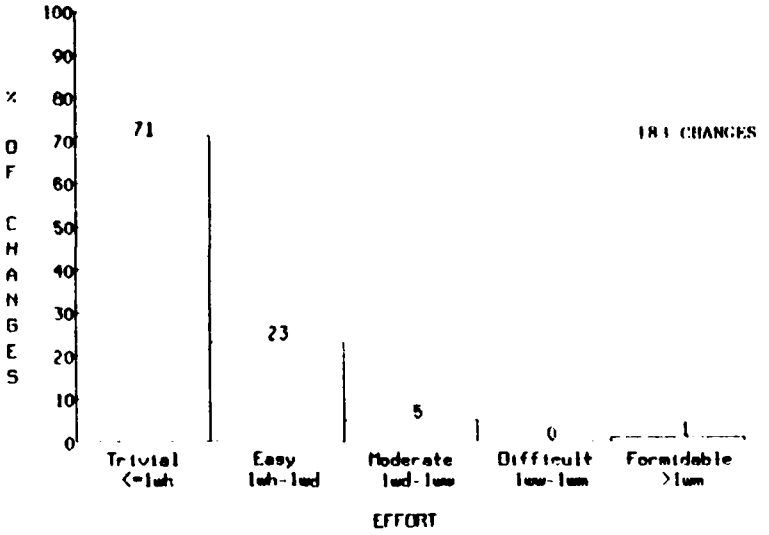


Fig. 4 — Effort to design changes excluding clerical errors and change completions (1978 — 1981)

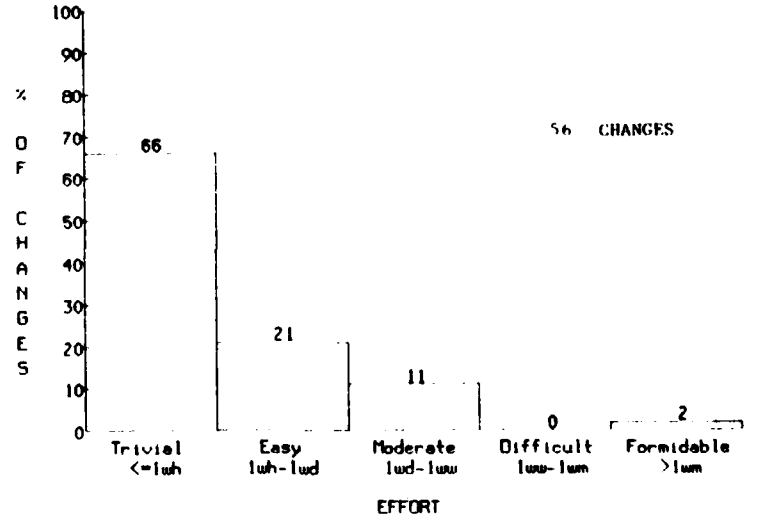


Fig. 5 — Effort to design changes excluding clerical errors and change completions (1978 — 1979)

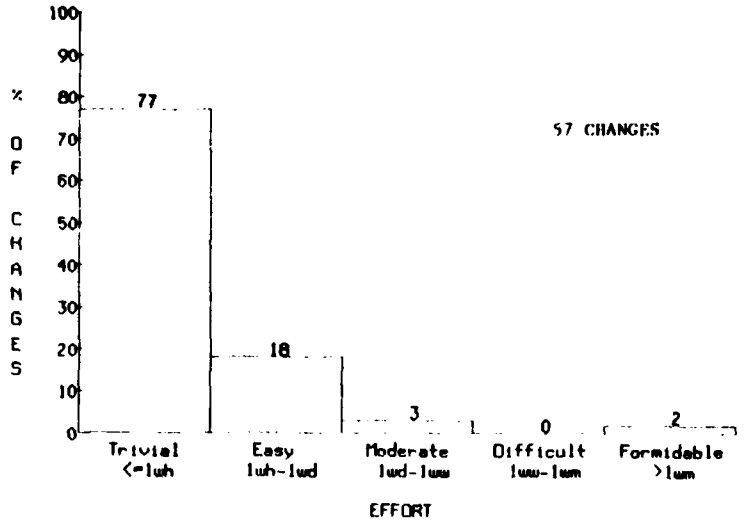


Fig. 6 — Effort to design changes excluding clerical errors and change completions (1981)

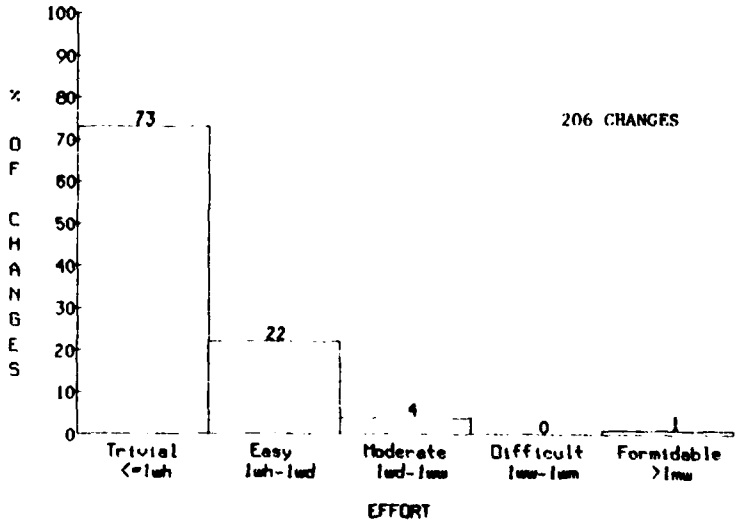


Fig. 7 — Effort to design changes excluding clerical errors (1978 — 1981)

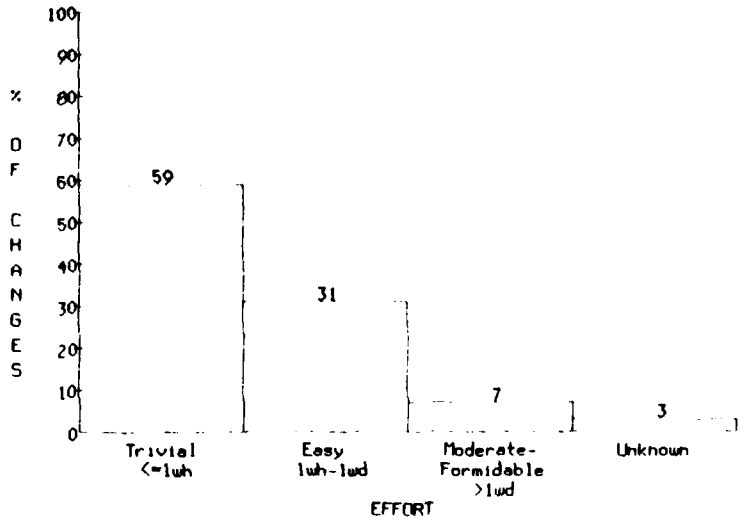


Fig. 8 — SEL1 effort to design changes (Fig. 5.8 from Weiss 1981)

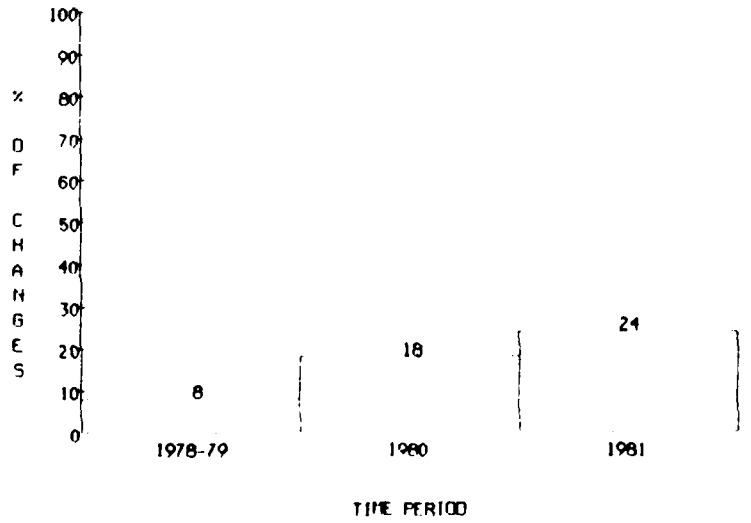


Fig. 9 — Percentage of changes requiring examination of more sections than those changed (1978 — 1981)

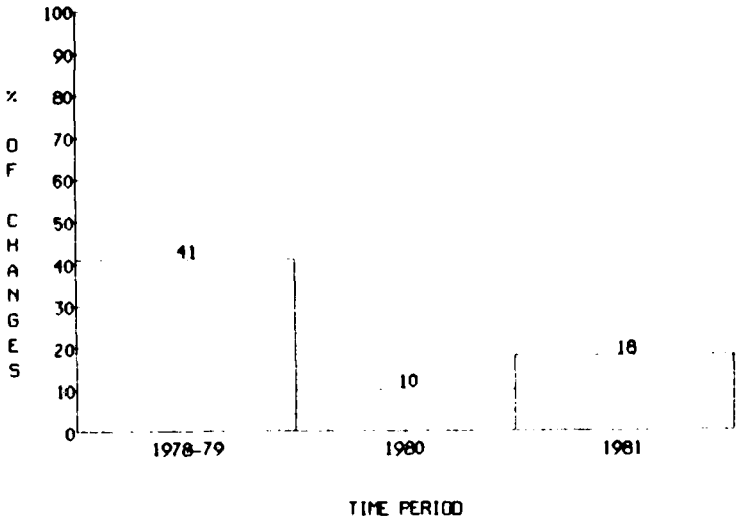


Fig. 10 — Percentage of changes involving more than one section (1978 — 1981)

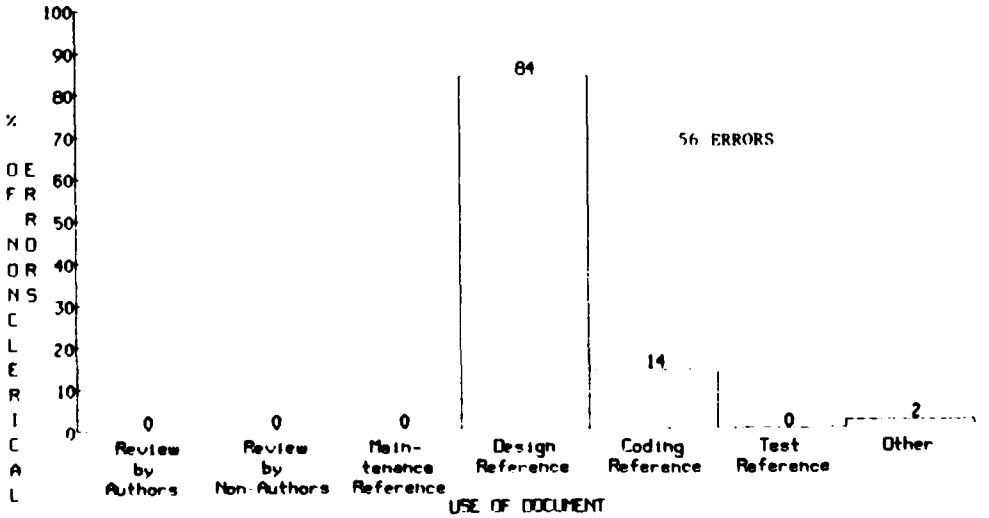


Fig. 11 — Detection of nonclerical errors (1981)

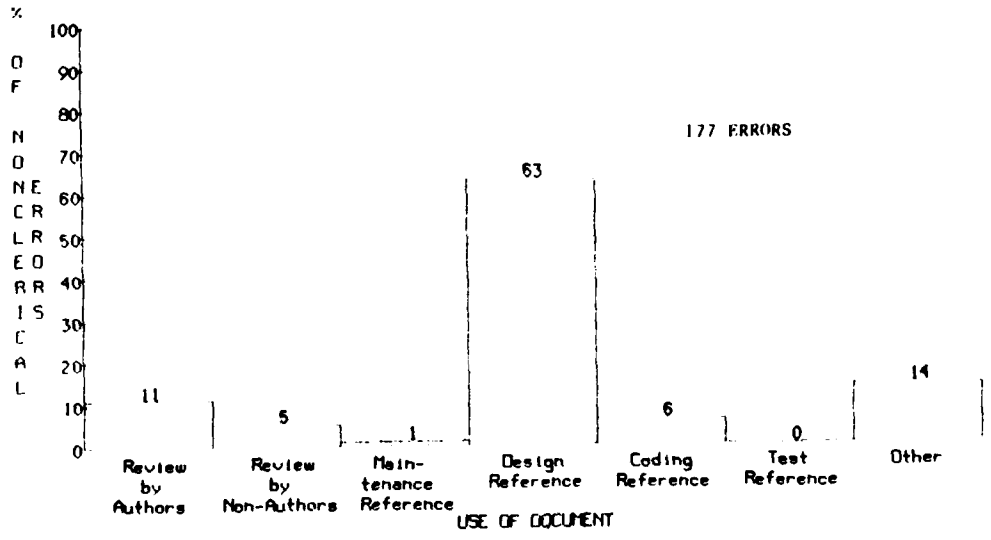


Fig. 12 — Detection of nonclerical errors (1978 — 1981)

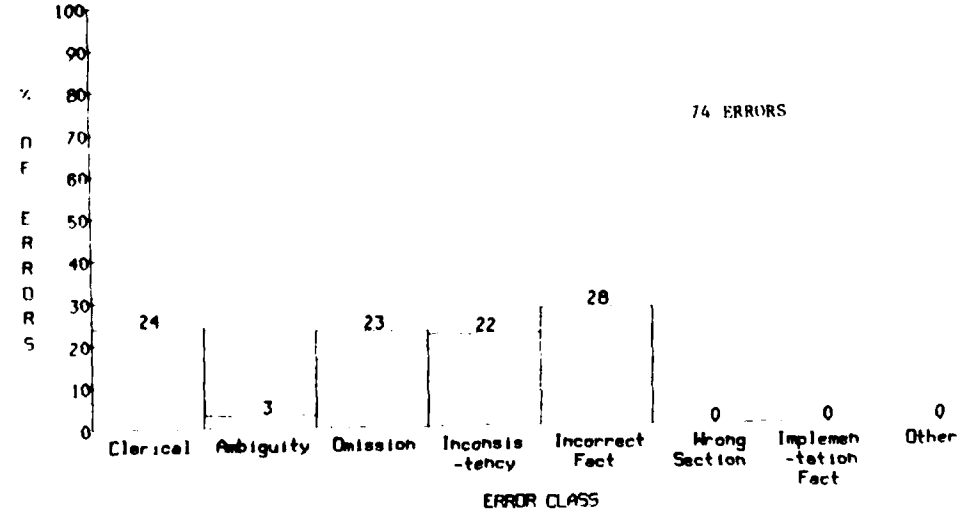


Fig. 13 — Error classes (1981)

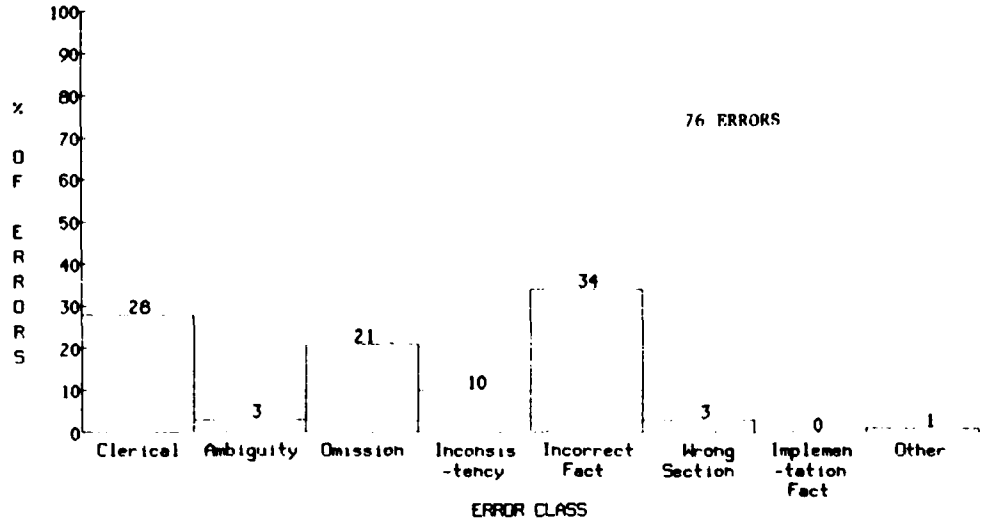


Fig. 14 — Error classes (1978 — 1979)

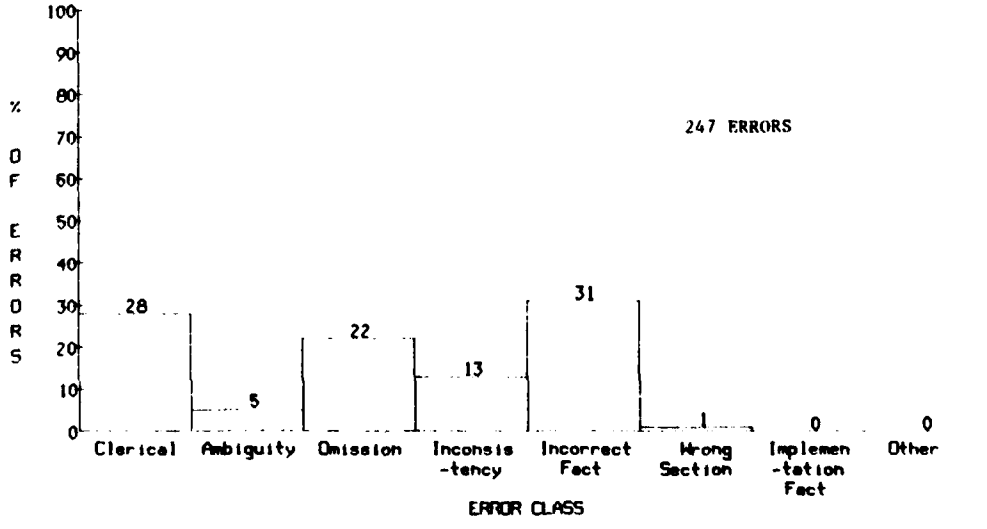


Fig. 15 — Error classes (1978 — 1981)

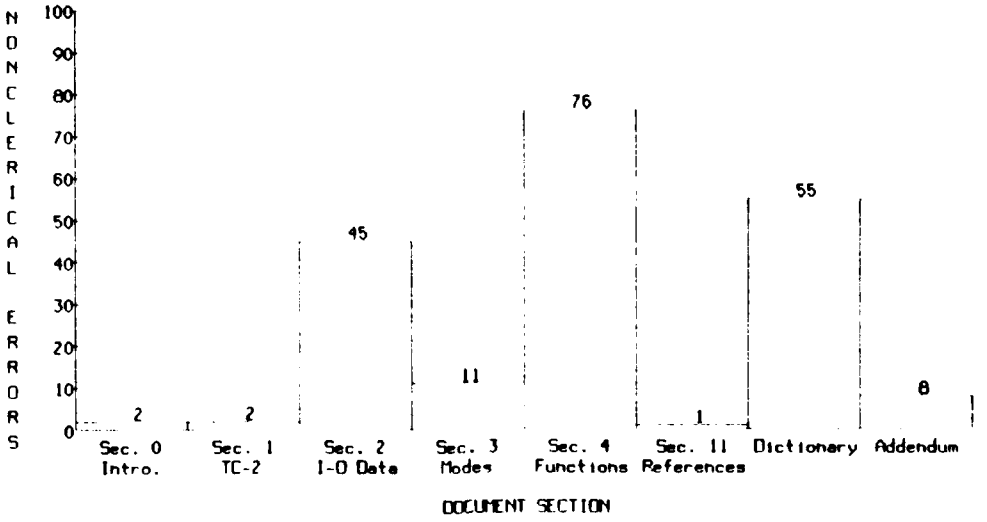


Fig. 16 — Nonclerical errors affecting major SRD sections (1978 — 1981)

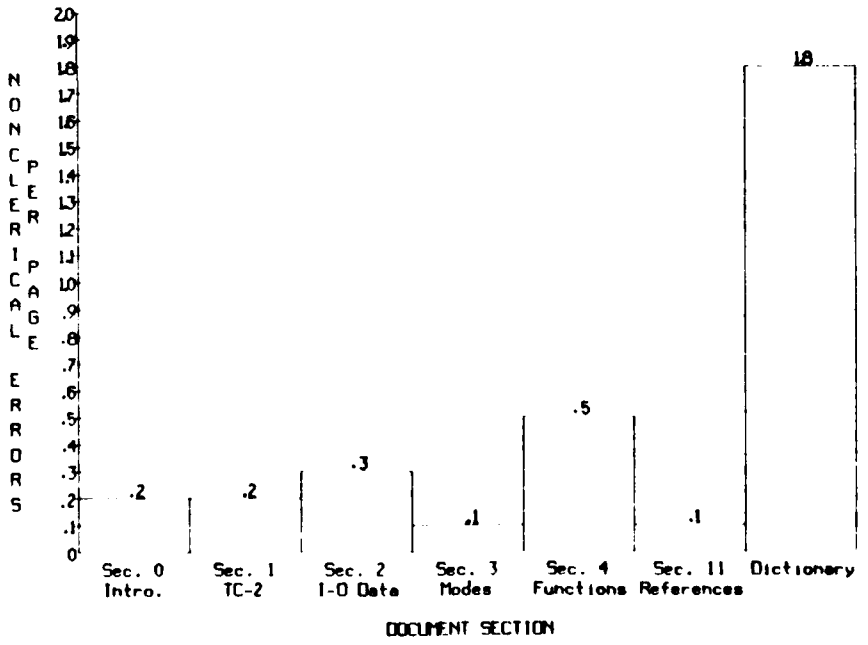


Fig. 17 — Error-per-page ratios for major SRD sections (1978 — 1981)

ACKNOWLEDGMENTS

Special thanks go to Paul Clements who patiently answered our seemingly endless questions about submitted CRFs. Miss Tamara Lewis is responsible for the high-quality histograms, which she produced despite problems with our home-grown plot package. Of course, we owe much to all those who have patiently filled out CRFs and who continue to do so.

REFERENCES

Basili, Victor R., and Weiss, David M. 1981.
Evaluation of a Software Requirements Document By Analysis of Change Data.
Proceedings, Fifth International Conference on Software Engineering, pp. 314-323. Long Beach CA: IEEE Computer Society.

Bell, Thomas E.; and Thayer, Thomas A. 1976.
Software Requirements: Are They Really A Problem?
Proceedings, 2nd International Conference On Software Engineering, pp 61-68. Long Beach CA: IEEE Computer Society.

Britton, Kathryn H., and Parnas, David L. 1981.
A-7E Software Module Guide.
NRL Memorandum Report 4702. Washington, DC: Naval Research Laboratory.

Britton, Kathryn H.; Parnas, David L.; and Weiss, David M. 1982.
Interface Specifications For The SCR (A-7E) Extended Computer Module.
NRL Memorandum Report. Forthcoming. Washington DC: Naval Research Laboratory.

Clements, Paul C. 1981.
Function Specifications for the A-7E Function Driver Module.
NRL Memorandum Report 4658. Washington DC: Naval Research Laboratory.

------. 1982.
Interface Specifications for the A-7E Shared Services Module
NRL Memorandum Report. Forthcoming. Washington DC: Naval Research Laboratory.

Dijkstra, Edsger W. 1968.
Cooperating Sequential Processes.
Programming Languages, ed. F. Genuys, pp. 43-112. New York: Academic Press.

Fryer, Sandra R., and Weiss, David M. 1981.
Evaluation of the A-7E Software Requirements Document By Analysis of Change Data: Two Years of Change Data.
Paper presented at the 15th Annual Asilomar Conference On Circuits, Systems, and Computers, November 1981.

Heninger, Kathryn L. 1980.
Specifying Software Requirements for Complex Systems: New Techniques and Their Application.
IEEE Transactions on Software Engineering, vol SE-6, no 1, January 1980.

Heninger, Kathryn L.; Kallandar, John; Parnas, David L.; and Shore, John E. 1978.
Software Requirements for the A-7E Aircraft.
NRL Memorandum Report 3876. Washington, DC: Naval Research Laboratory.

Hoare, C. A. R. 1974.
Monitors: An Operating System Structuring Concept.
Communications of the ACM, vol. 17, no. 10 (October 1974), pp. 549-557.

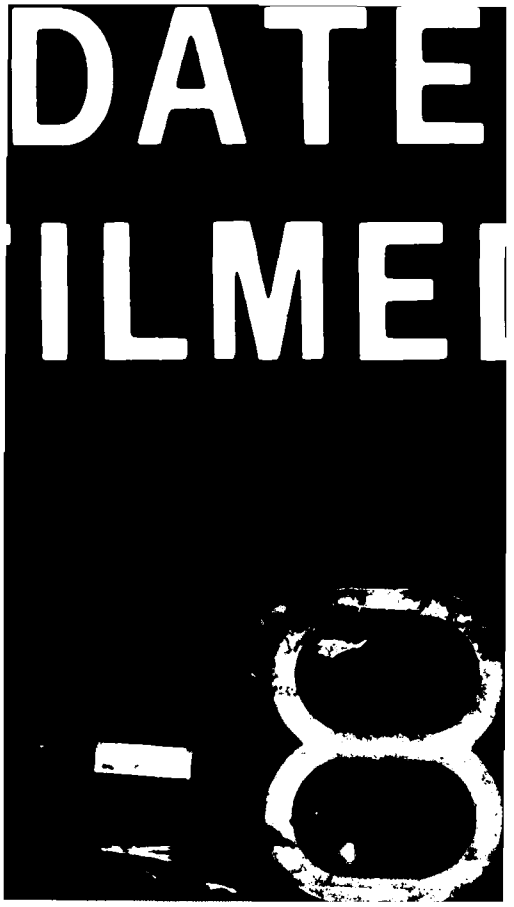
Parker, Robert A.; Heninger, Kathryn L.; Parnas, David L.; and Shore, John E. 1980.
Abstract Interface Specification for the A-7E Device Interface Module.
NRL Memorandum Report 4385. Washington DC: Naval Research Laboratory.

Parnas, David L. 1972.
On the Criteria To Be Used in Decomposing Systems into Modules.
Communications of the ACM, vol. 15, no. 12 (December 1972) pp. 1053-1058.

------. 1977.
Use of Abstract Interfaces in the Development of Software for Embedded Computer Systems.
NRL Report 8047. Washington, DC: Naval Research Laboratory.

Shooman, M. L., and Bolsky, M. I. 1975.
Types, Distribution, and Test and Correction Times For Programming Errors.
Proceedings -- 1975 International Conference on Reliable Software.
SIGPLAN Notices, vol 10, no 6 (6 June 1975) pp 347-357.

Weiss, David M. 1981.
Evaluating Software Development By Analysis Of Change Data.
Ph.D. Dissertation. Technical Report TR-1120. College Park: University of Maryland.


DATE
ILME